A VLSI Design for a Systolic Viterbi Decoder

T. K. Truong and E. Satorius
Communications Systems Research Section

M. T. Shih and I. S. Reed
Department of Electrical Engineering, University of Southern California

A systolic Viterbi decoder for convolutional codes is developed. This decoder uses the trace-back method to reduce the amount of data needed to be stored in registers. It is shown that this new algorithm requires a smaller chip size and achieves a faster decoding time than other existing methods.

I. Introduction

Convolutional coding with Viterbi decoding [1] is a powerful method for forward error correction. As a consequence there is a growing need for implementing the Viterbi decoder in VLSI in deep-space communication [2].

There are two classes of algorithms established for realizing the Viterbi decoder. These are the register-exchange and trace-back methods [3]. Unfortunately, each class has drawbacks. Both require a substantial amount of storage space and hardware for even a moderate speed while the latter, for long constraint lengths, also needs a long decoding time.

In this article, the Viterbi decoding algorithm is first presented for comparison with other methods. For this example, a modified trace-back algorithm is created and shown to require a minimal number of storage devices and a short decoding time. Next, a systolic architecture is developed for this modified trace-back decoding algorithm. It is demonstrated for this new trace-back architecture that the tracking, updating, and storing of the hypothesized information sequences for the Viterbi decoder can be accomplished simultaneously during a single clock cycle. Finally, a suitable VLSI implementation is suggested for this new systolic architecture.

II. Example of a Viterbi Decoder for a (3,1/2) Convolutional Code

Let K and R be the constraint length and the rate, respectively, of what is denoted a (K, R) convolutional code (CC). In this section, the nature of the Viterbi decoding algorithm is demonstrated by using a (3,1/2) convolutional code.

Consider the example of a (3,1/2) CC with the generator polynomial $G(x) = (x^2 + x + 1, x^2 + 1)$. The encoder for this code is shown in Fig. 1 where one observes that the encoder is composed of a two-stage shift register $A = (A_1, A_0)$ with three modulo-2 adders and a multiplexer for converting a parallel to a serial output, where A_i for (i = 1, 2) denotes a one-bit register. Let the information sequence be $\underline{u} = (u_1, u_2, u_3, u_4, \ldots) = (0, 0, 1, 0, \ldots)$. Af-

ter encoding the information sequence $\underline{u} = (0, 0, 1, 0, \ldots)$, the output code word is given by $\underline{v} = (v_1, v_2, v_3, \ldots) = (00, 00, 11, 10, \ldots)$.

The trellis diagram for a particular $(3, 1/2)$ CC is shown in Fig. 2. In each column of the trellis there are $2^K = 2^2 = 4$ states of the shift register. These are the four possible states of the shift register of the encoder. Each information bit causes the shift register to change state. This is represented by a branch from the present node to the next node. Each branch in the jth column is labeled with the single output code-word frame v_j. The upper branch leaving a node at time unit $j-1$ represents the "zero" input bit $u_j = 0$, while the lower branch represents the "one" input bit $u_j = 1$. The code word that corresponds to the information sequence $\underline{u} = (0, 0, 1, 0, 1, 0, 0, 0, \ldots)$ is shown in Fig. 2 as a heavy-line path.

Assume that a code word $\underline{v} = (v_1, v_2, v_3, \ldots)$ is transmitted over a binary symmetric channel (BSC) and that the received sequence is $\underline{r} = (r_1, r_2, r_3, \ldots)$. The branch metric from state S_l at time unit $j-1$ to S_k at time unit j is defined by

$$d_{j-1,j}(l,k) = \|r_j - v_j\| \tag{1}$$

where $d_{j,j-1}(l,k)$ denotes Hamming path distance from state S_l at time unit $j-1$ to S_k at time unit j. Here also $\| r_j - v_j \|$ is the Hamming weight of the difference of the two binary vectors r_j and v_j. The partial path metric for a state S_k up until the end of the first j branches of a path is denoted by $M_j(k)$. If there is a state S_l at time $j-1$ which can change to state S_k at time j along the given path, then the partial path metric is expressible by the difference equation

$$M_j(k) = M_{j-1}(l) + d_{j,j-1}(l,k) \tag{2}$$

where $M_0(k) = 0$ for $k = 0$. The smallest metric for all paths terminating at state S_k at time j is given by

$$P_j(k) = \min\ M_j(k) \tag{3}$$

where the minimum is over all possible partial-path metrics that end at state S_k at time j. $P_j(k)$ is the weight of what is called the survivor path to state S_k at time unit j.

The Viterbi decoder for a convolutional code finds the survivor path, the path of minimum metric, which reaches a given state at time j. This survivor path is dependent on the input information bits so that the decoded bits are read off easily along this path.

The Viterbi algorithm is outlined as follows:

(1) Beginning at time unit $j = 1$, compute the partial metric for the single path entering each state. Store the path (survivor) and its metric for each state.

(2) Increase j by 1. Compute the partial metric for all the paths entering a state by adding the branch metric entering that state to the metric of the connecting survivor at the preceding time unit. For each state, store the path with the smallest metric (the survivor), together with the metric, and eliminate all other paths.

(3) If $j < L_c$, repeat step (2). Otherwise, stop.

In the above, L_c is the length of the code word. The final survivor with the smallest metric can be used to decode the information bit along this path.

As an example of the above Viterbi algorithm, assume that a code word of this $(3, 1/2)$ CC is $\underline{v} = (v_1, v_2, v_3, \ldots) =$ (00, 00, 11, 10, 00, 10, 11, 00, 11, 01, 01, 00, 10, 11, 11, 10, 11, 11, 01, 01) and this code word is transmitted over a BSC. The received sequence is $\underline{r} = (r_1, r_2, r_3, \ldots) =$ (00, 01, 11, 10, 10, 10, 11, 00, 11, 11, 01, 00, 10, 10, 11, 10, 11, 11, 01, 01). The Viterbi decoder for this convolutional code is illustrated in Fig. 3. At the beginning of the trellis diagram, one observes from Fig. 3 that only a single path enters each state. For example, states S_0 and S_2 follow from state S_0 during the first time unit. By step (2), the partial path metric for state S_0 at time unit 1 is $M_1(0) = d_{0,1}(0,0) = \| r_1 - v_1 \| = 0$. Thus the survivor path has metric $P_1(0) = M_1(0) = 0$. Similarly, one obtains the partial metric for state S_2 as $P_1(2) = M_1(2) = 2$.

At the second time unit, only single paths enter states S_0, S_2, S_1, and S_3. Thus the weights of survivors at $j = 2$ are found from $P_2(0) = M_2(0) = M_1(0) + d_{1,2}(0,0) = 1$, $P_2(2) = M_2(2) = M_1(0) + d_{1,2}(0,2) = 1$, $P_2(1) = M_2(1) = M_1(2) + d_{1,2}(2,1) = 4$, $P_2(3) = M_2(3) = M_1(2) + d_{1,2}(2,3) = 2$.

At time unit 3, there are two branches entering each state. One of these two paths entering state S_0 is obtained by the algorithm using the following results: First $M_3(0) = M_2(0) + d_3(0,0) = 3$ or $M_3(0) = M_2(1) + d_3(1,0) = 4$; hence, $P_3(0) = \min\{3, 4\} = 3$ for state S_0; similarly, one yields $P_3(2) = 1$ for state S_2, $P_3(1) = 2$ for state S_1, and $P_3(2) = 2$ for state S_3.

For the remaining frame times, the same procedure yields the survivor path segments from each partial path metric for each state. In Fig. 3, the metrics of the survivors

are denoted at each node. Assume that input information bits stop at time unit 20 and that one chooses the smallest metric among the four nodes. Thus, at time unit 20, the smallest metric among survivors is $P_{20}(1) = 4$. This survivor path that reaches state S_1 at time unit 20 is heavy lined in Fig. 3. For this path, the decoded information bits are $(0,0,1,0,1,0,0,0,1,1,0,1,0,0,1,0,0,1,1,0)$.

Define W_i to be a window of length L at time unit i, where the times of the start and end of the window are i and $i+L-1$, respectively. It is shown in [4] that if the length of the information bit stream is large, the decoding-window length L is usually several times the constraint length. At time unit L, the decoder chooses the surviving path that reaches the state with the smallest survivor by the first decoding window W_1. The first branch in W_1 can be decoded as the first information bit. This decoding window then shifts one time unit to be the next decoding window W_2. This new decoding window can be used to decode the second decoded information bit. The following section discusses two different methods for realizing the Viterbi decoder for a decoding window of length $L = 5K$.

III. Methods for Realizing the Viterbi Decoder

There are two methods for appoximately realizing the Viterbi decoder [3]. The first method, called the register-exchange method, calls for all paths to be stored for each of the 2^K states. At each time unit, a new branch is processed by comparing the partial path metrics. Then certain registers are interchanged corresponding to the paths that survived the comparison, and a new information bit is added at one end of each register. After $5K$ branches have been processed, the first bit of register, corresponding to the smallest survivor, is shifted out as the first decoded bit. The register exchange algorithm is illustrated in [5]. This algorithm for long constraint-length code requires a substantial amount of storage space and hardware for even moderate decoding speeds.

The second method, called the trace-back method, does not store the actual information sequence but instead stores the results of each comparison. After $5K$ branches have been processed, the trellis connections are recalled in reverse order. That path traced back through the trellis diagram is used to decode the first bit. The following constitutes the trace-back algorithm.

To trace back a survivor path through the trellis diagram, one needs to store the trellis connections for each state at each time unit. For example, the survivor paths from time unit 3 to time unit 4 in Fig. 3 are shown in detail in Fig. 4(a). At time unit 3, the state transitions to state S_0, S_2, S_1, and S_3 at time unit 4 are the states S_1, S_1, S_2, and S_3, respectively. In Fig. 4(a), if one needs to trace the path from S_2 at time unit 4 back to S_1 at time unit 3, then one needs the information that a state transition from S_1 at time unit 3 to state S_2 occurred at time unit 4. Note that the last bit of state vector S_1, i.e., the last bit of 01, shifts out of the shift register A prior to the time that state S_1 changes to S_2. Thus this last bit of state vector S_1 needs to be stored in order to trace state S_2 back to S_1. Let $y_j(k)$ be this one-bit information needed to trace state S_k at time unit j back to the state of the survivor at time unit $j-1$. By this definition, $y_4(2) = 1$. As shown in Fig. 4(a), the values $y_4(0)$, $y_4(2)$, $y_4(1)$, and $y_4(3)$ are obtained as 1, 1, 0, and 1, respectively.

To trace a state vector back to its previous state vector survivors, one notes first by the example in Fig. 4(a) that the current state S_2 at time unit 4 is the result of shifting a 1 into register A. This bit is the first bit of state vector 10 of S_2 and should be deleted when traced back. Also the $y_4(2)$ should be linked to the last bit of this state vector. Thus one can determine the previous state vector to be 01 by deleting the first bit of state vector 10 of S_2 and concatenating with the $y_4(2)$, which is bit "1". Every previous state of states S_0, S_2, S_1, and S_3 can be easily obtained by using the same method. These previous states are S_1, S_1, S_2, and S_3, respectively; they are shown in Fig. 4(b).

As mentioned in Section II, one can use each decoding window W_i to decode the ith decoded bit. Note that the decoding window length $L = 5K = 10$ in our example. To decode the first decoded bit, first one uses the first decoding window W_1 to construct the survivors by the Viterbi algorithm in Section II. Also one obtains the $y_i(k)$ for $1 \leq i \leq 10$. Let $\underline{y}_i$ be the column vector with elements $y_i(k)$ where $k = 0,2,1,3$, i.e., $\underline{y}_i = [y_i(0), y_i(2), y_i(1), y_i(3)]^T$. One obtains the $\underline{y}_i$ for $1 \leq i \leq 10$ to be $\underline{y}_1 = [0,0,X,X]^T$, $\underline{y}_2 = [0,0,0,0]^T$, $\underline{y}_3 = [0,0,0,0]^T$, $\underline{y}_4 = [1,1,0,1]^T$, $\underline{y}_5 = [1,1,0,1]^T$, $\underline{y}_6 = [0,0,0,0]^T$, $\underline{y}_7 = [1,0,1,1]^T$, $\underline{y}_8 = [0,1,0,0]^T$, $\underline{y}_9 = [0,0,0,0]^T$, and $\underline{y}_{10} = [1,0,0,0]^T$. In the above, the $\underline{y}_1 = [0,0,X,X]^T$ means that there are not any state changes to state S_1 or S_3 at time unit 1. Among these survivors, one chooses the state vector with the smallest metric of the survivors at time unit 10 to be $X = 11$ (state S_3). These $\underline{y}_i$s and state vector $X = 11$ are shown in Fig. 5(a). In Fig. 5(a), the elements in each column $\underline{y}_i$ represent the associated $y_i k$ for $k = 0,2,1,3$. One then recursively traces the survivor from state vector 11 at time unit 10 back to time unit 1 using the method shown in Fig. 4(b). First, at time unit 9, the state vector is obtained as $DMSB(11) * y_{10}(3) = 1 * 0 = 10$ where

$DMSB(11)$ denotes the state vector 11 without the first bit and the operation $*$ denotes the concatenation operation. Then at time unit 8, the state vector is obtained as $DMSB(10) * y_9(2) = 0 * 0 = 00$. Also, the state vector at time unit 7, 6, 5, 4, 3, 2, 1 can be obtained sequentially as 00, 01, 10, 01, 01, 10, 00. Finally, one obtains the first decoded bit as $Z = MSB(00) = 0$ where $MSB(00)$ denotes the first bit of 00 since the input information bit is the first bit of this state vector at time unit 1.

To decode the second decoded bit, the second decoding window W_2 is needed. Thus one needs the survivor at time unit 11. The state vector with the smallest metric at time 11 is $X = 01$ and $\underline{y}_{11} = [1, 1, 1, 0]^T$, which are shown in Fig. 5(b). Using the same method above, one can trace state vector 01 at time unit 11 back to time unit 2, shown sequentially, and one obtains the state vector $X = 00$ at time unit 2. Thus the second decoded bit is $Z = MSB(00) = 0$.

The same procedure is used again to decode the following decoded bit. This procedure stops when the time unit is equal to $L_c + 5K + 1$ where L_c is the length of the code word.

The trace-back algorithm is summarized as follows. Note that because the start state is S_0 at time unit 0, one assigns metric $M_0(k) = \infty$ for $k \neq 0$.

(1) Initially let $M_0(0) = 0$ and $M_0(k) = \infty$ for $k \neq 0$.

(2) For each $k = 0, 1, 2, 3$, find an $l \in \{0, 1, 2, 3\}$ such that $M_{j-1}(l) + d_j(l, k)$ is minimum. Then

$$M_j(k) = M_{j-1}(l) + d_j(l, k)$$

$$y_j(k) = LSB(l)$$

(3) If $j < 5K$, go to (2); otherwise, find an $m \in \{0, 1, 2, 3\}$ such that $M_j(m)$ is minimum. Then

$$X = m$$

Also, for $i = j, j-1, \ldots, j-5K+1$,

$$X = DMSB(X) * y_i(X)$$

Then

$$Z = MSB(X)$$

(4) If $j = L_c + 5K$, stop; otherwise, if $j \geq L_c$, $j \leftarrow j + 1$ and go to (3); otherwise, $j \leftarrow j + 1$ and go to (2).

$MSB(k)$ denotes the first bit of the binary representation of k. $LSB(l)$ denotes the last bit of binary representation of l. $DMSB(X)$ denotes the sequence of bits of state vector X without the first bit. L_c is the length of the code word. Step (2) is used to compute the partial path metric and store the information for choosing each survivor. Step (3) is used to trace back the survivor to find the decoded bit. First, the state vector of the state with the smallest metric of the survivor is assigned to be X when $j \geq 5K$. Then one can use this state vector to trace back the survivor by the data $\underline{y}_i$. Once the trace-back procedure is finished, the decoded bit is denoted by Z. Note that in step (2), one needs $(5K \times 2^K)$-bits storage space to store $y_j(k)$. Also in step (3), the trace-back procedure takes about $5K$ cycles for each decoded bit. This method requires a long decoding time.

IV. The Systolic Viterbi Decoder

The advantage of this systolic Viterbi decoder is that the trace-back operation is accomplished by processing a systolic array of registers in a pipeline fashion instead of waiting for the whole trace-back procedure. As a consequence, this systolic structure reduces the decoding time. However, if one traces back the survivor from time j to $j-1$, simultaneously the survivor is selected forward from time j to $j+1$. Thus one needs extra storage space to store the information needed to choose the survivor path at time unit j. As a result, one needs about twice the amount of storage space to store $y_j(k)$ as that needed in the original trace-back algorithm. This is expressed in detail in the following description of the operation of the systolic Viterbi decoder.

The systolic structure of the trace-back algorithm is illustrated in Fig. 6. As shown in Fig. 6, this systolic-array structure consists of a selection unit and $10K - 1 = 19$ path units. The selection unit processes step (2) of the trace-back algorithm, i.e., at time unit t, it computes the metrics of survivors of all nodes, selects the state vector with the smallest metric, and stores the information $y_t(k)$ for selecting each survivor. Note that this selection unit operates recursively. The inputs of the selection unit are the received sequence r_t and the metrics of the previous survivors $P_{t-1}(l)$. The outputs are the metrics of the survivors $P_t(l)$, the information for selecting survivors, and the state vector m of the state with the smallest metric.

Step (3) of the trace-back algorithm is modified and implemented by $10K - 1 = 19$ path units in a pipeline manner. Each path unit consists of 4-bit registers Y_i and 2-bit registers X_i for odd i or only a 4-bit register Y_i for

even i. The register Y_1 is used to store the elements of column vector $\underline{y}_j$ at time unit j, and the register Y_i for $1 < i \leq 19$ is used to store the data shifted from register Y_{i-1}. The register X_1 is used to store the state vector m of the state with the smallest metric, and the register X_i for $i = 3,5,\ldots,19$ is used to store the data shifted from register X_{i-2}; the one-bit register Z is used to store the decoded bit. The Register Transfer Language (RTL) developed in [6] is used to describe in detail the operation of this systolic structure as shown in Fig. 6. At time unit t, the contents in each register Y_i for $i = 1,2,\ldots,18$ is transferred to the following register Y_{i+1} and the $y_t(k)$s for $k = 0,1,2,3$, which are the output of selection units, are stored in the register Y_1. If $t \geq 5K$, the contents in register X_i without the first bit for odd $i \leq 17$ is concatenated with the corresponding element in register Y_i, where the address of this element is the contents of X_i, to generate a new state vector. This new state vector is then transferred to the following register X_{i+2}. Also, the state vector m of the state with the smallest metric is transferred to the register X_1, and the first bit of the contents of register X_{19} is stored in register Z as the decoded bit.

Using the same example as that in Section III, one obtains the contents in each register at different time units as shown in Fig. 7. Note that at time unit $10K = 20$, one obtains the first decoded information bit stored in register Z and then one after each time unit. One obtains the decoded information bits sequentially. Each odd path unit stores 2^k bits of data in Y_i and k bits in X_i, while each even path unit stores 2^K bits of data in Y_i.

Note also that it is possible to further reduce the number of registers. As shown in Fig. 7 at time unit 19, the first bit of X_{19} is the last bit of X_{17} at time unit 18. Thus the registers Y_{18}, Y_{19}, and X_{19} are not needed in this (3,1/2) Viterbi decoder, and register Z is directly connected to register Y_{17} to store the first bit of the contents of Y_{17}. Also, the first decoded bit is decoded at time unit 19 instead of time unit 20. If a structure is similar to that used in a (3, 1/2) convolutional code, the general systolic Viterbi decoder can be obtained.

It has been shown [3, 7] that one can use any state vector to trace back if the window length is equal to $5K$. This is due to the fact that all survivor paths most likely will merge within a window length of $5K$. Then the operation of selecting the state vector with the smallest metric of the survivor in the selection unit is eliminated. As a result, this tracking, updating, and storing of the hypothesized information sequence can be accomplished simultaneously during a single clock cycle. This systolic-array structure minimizes the interconnections between components. Hence, this new architecture is suitable for a VLSI implementation.

V. Conclusion

The realization of the systolic Viterbi decoder for convolutional codes is demonstrated. This fully parallel architecture of decoding requires a minimum amount of storage space and decoding time. This makes it possible to readily implement a Viterbi decoder with VLSI circuits.

References

[1] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*, New York: McGraw-Hill, 1979.

[2] R. L. Miller, L. J. Deutsch, and S. A. Butman, *On the Error Statistics of Viterbi Decoding and the Performance of Concatenated Codes*, JPL Publication 81-9, Jet Propulsion Laboratory, Pasadena, California, September 1, 1981.

[3] G. C. Clark, Jr. and J. B. Cain, *Error-Correction Coding for Digital Communications*, New York: Plenum Press, 1981.

[4] R. E. Blahut, *Theory and Practice of Error Control Codes*, Massachusetts: Addison-Wesley Publishing Company, May 1984.

[5] R. J. McEliece, *The Theory of Information and Coding*, Massschusetts: Addison-Wesley Publishing Company, 1977.

[6] T. C. Bartee, I. L. Lebow, and I. S. Reed, *Theory and Design of Digital Machines*, New York: McGraw-Hill Book Company, 1962.

[7] S. Lin and D. J. Costello, Jr., *Error Control Coding*, Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1983.

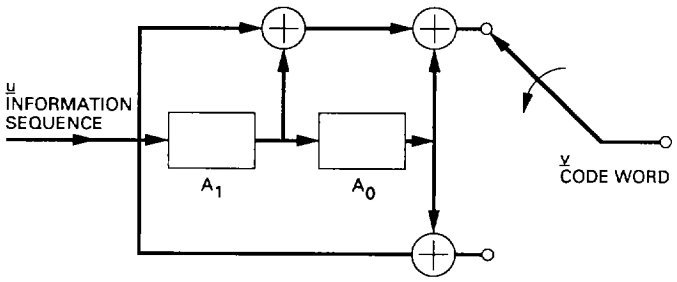

Fig. 1. A (3,1/2) convolutional code.

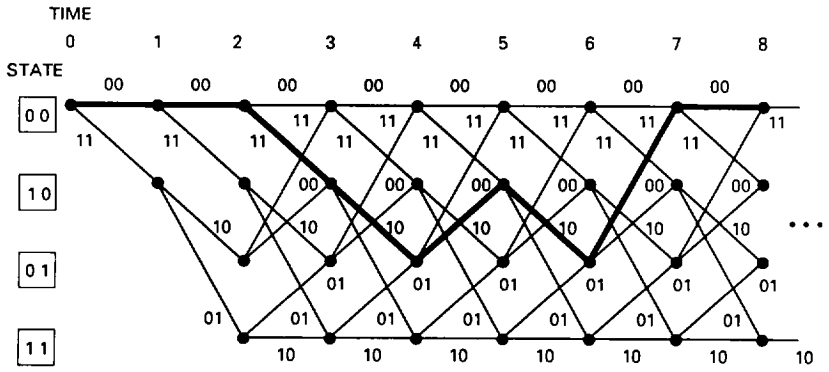

Fig. 2. Trellis diagram for a (3,1/2) convolutional code.

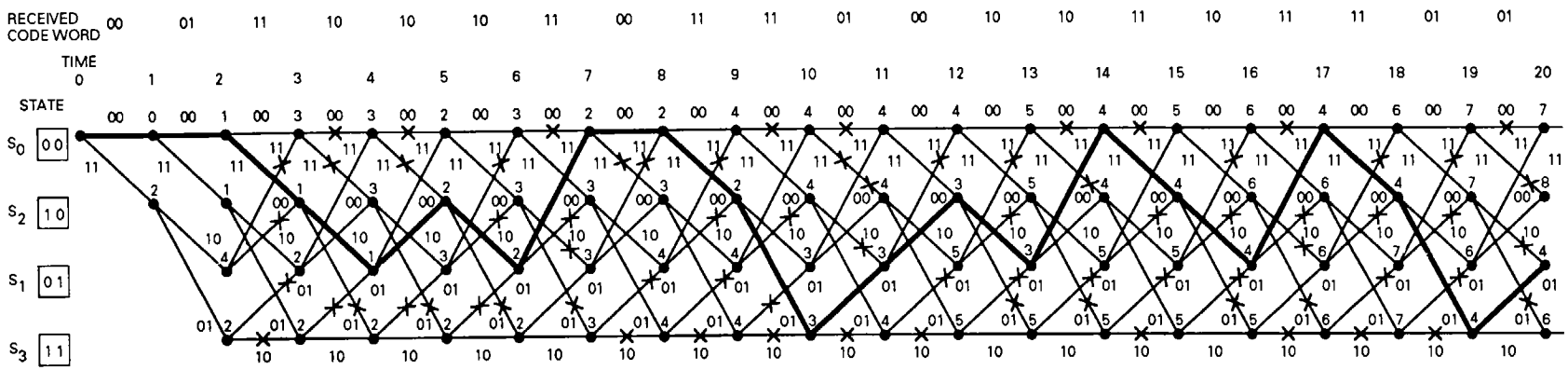

Fig. 3. Viterbi algorithm for a binary symmetric channel.

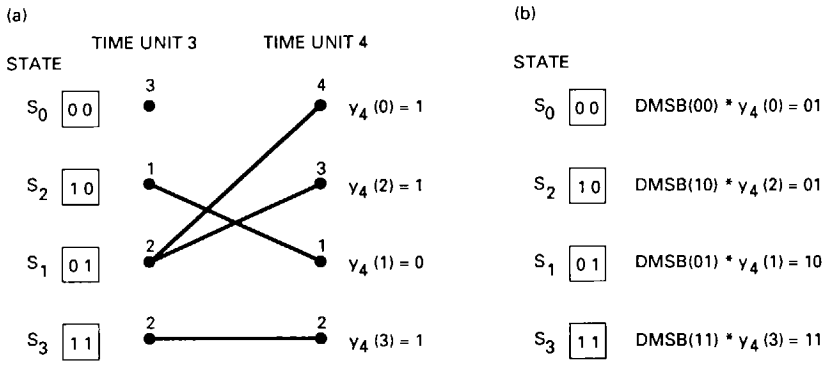

Fig. 4. Survivor paths from time unit 3 to time unit 4 and the previous states of S_0, S_2, S_1, and S_3: (a) the assignment of $Y_4(k)$ in the trace-back algorithm; (b) the traced-back state vector for each state.

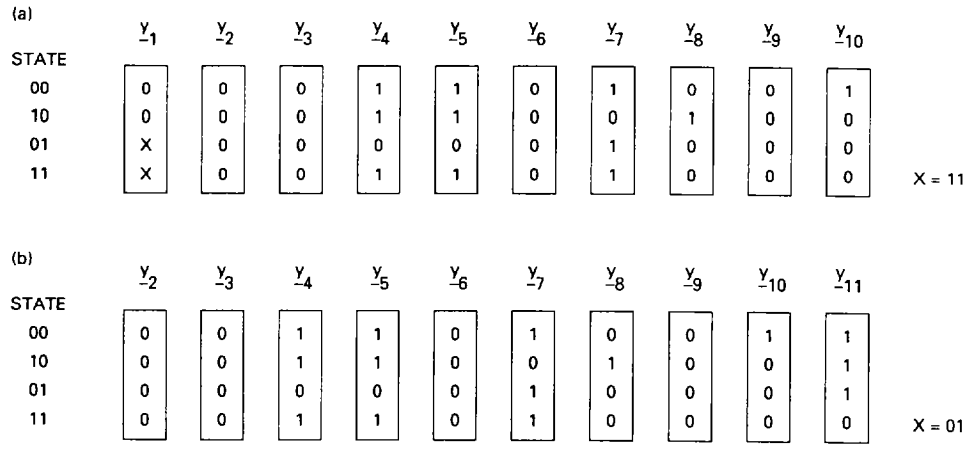

Fig. 5. The column vector y_i, the state vector X with the smallest metric: (a) the data of y_i in decoding window W_1 for decoding the first bit; (b) the data of y_i in decoding window W_2 for decoding the second bit.

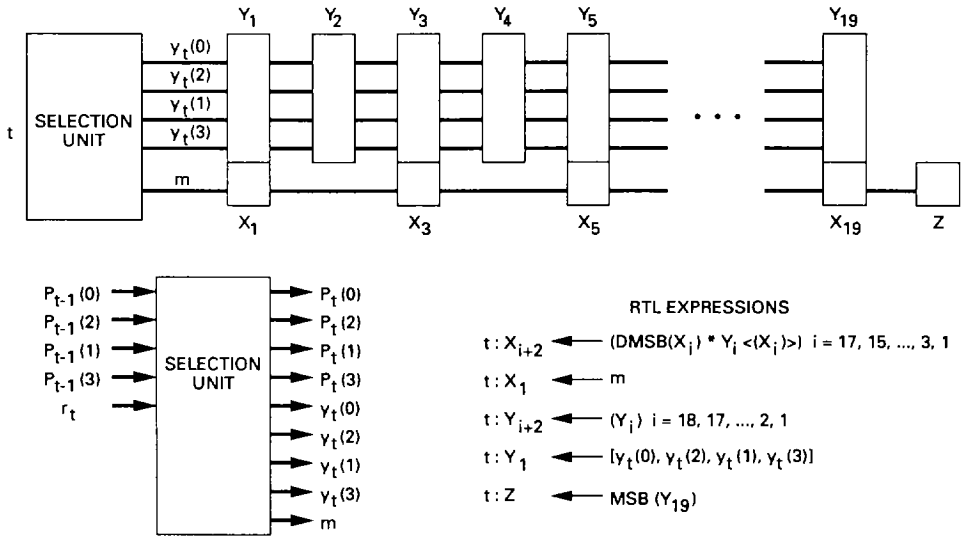

Fig. 6. The structure of the systolic Viterbi decoder.

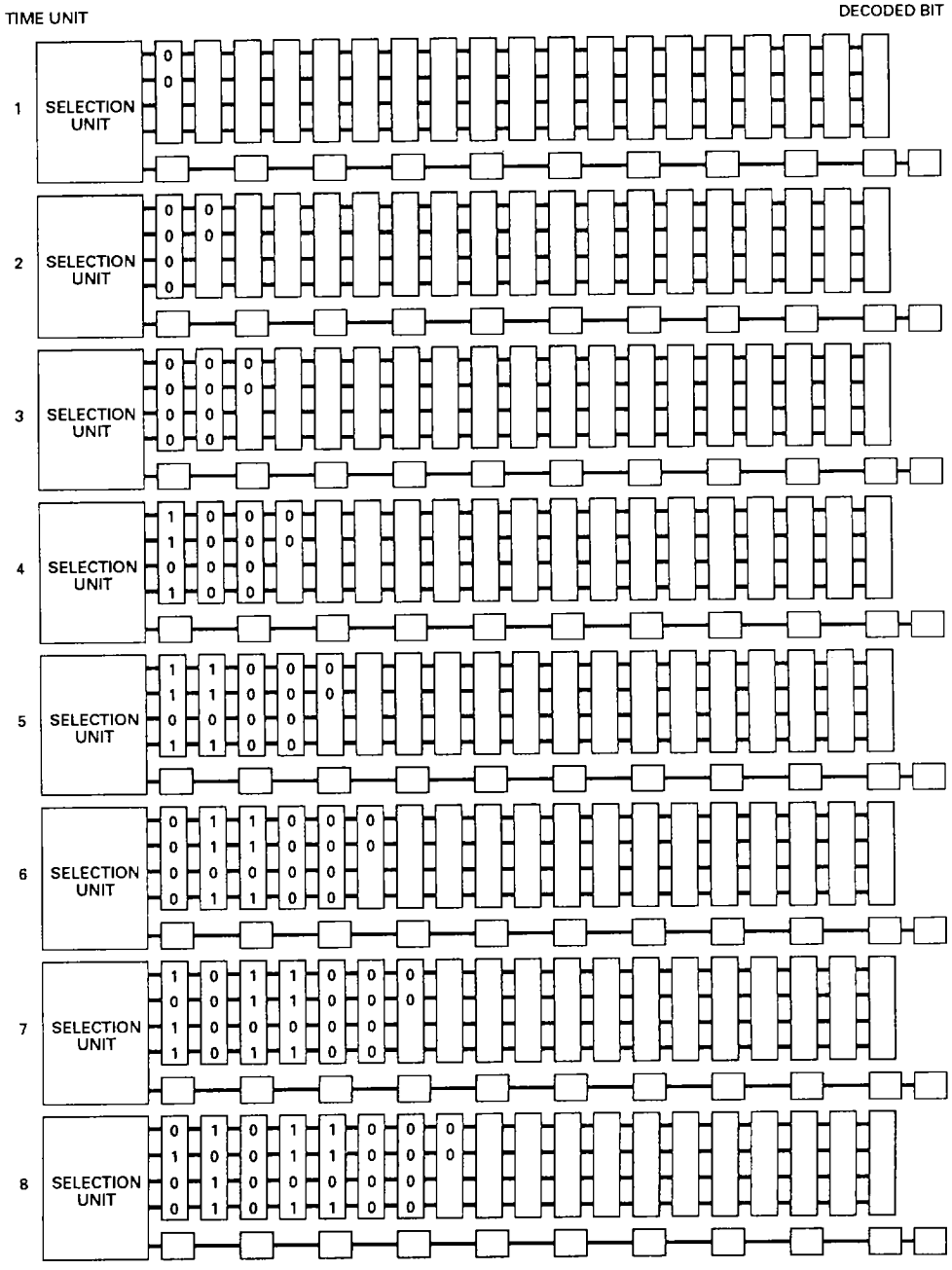

Fig. 7. The systolic Viterbi decoder for a (3,1/2) convolutional code.

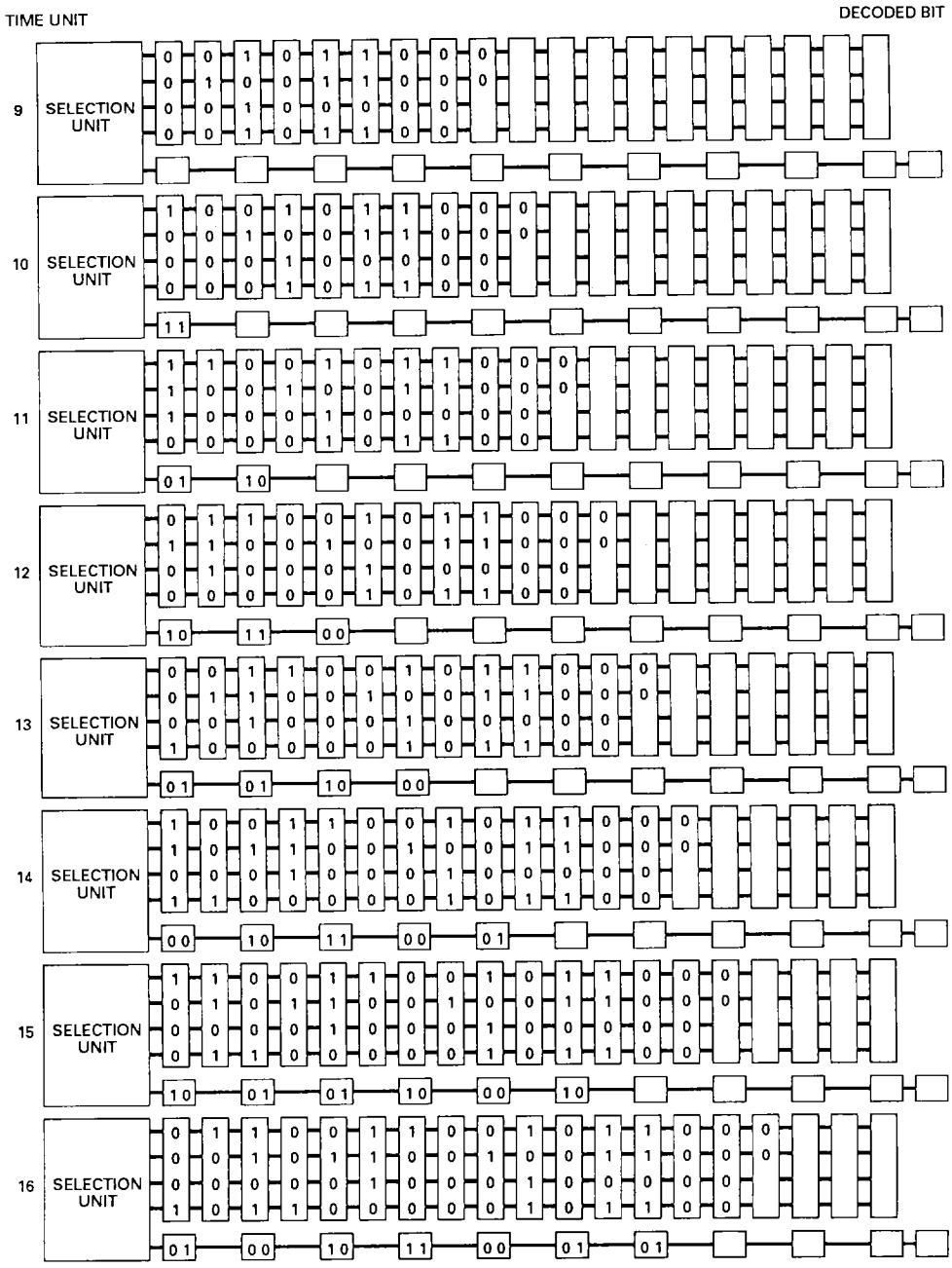
TIME UNIT
DECODED BIT
9
SELECTION UNIT
10
SELECTION UNIT
11
SELECTION UNIT
12
SELECTION UNIT
13
SELECTION UNIT
14
SELECTION UNIT
15
SELECTION UNIT
16
SELECTION UNIT

Fig. 7. (contd)

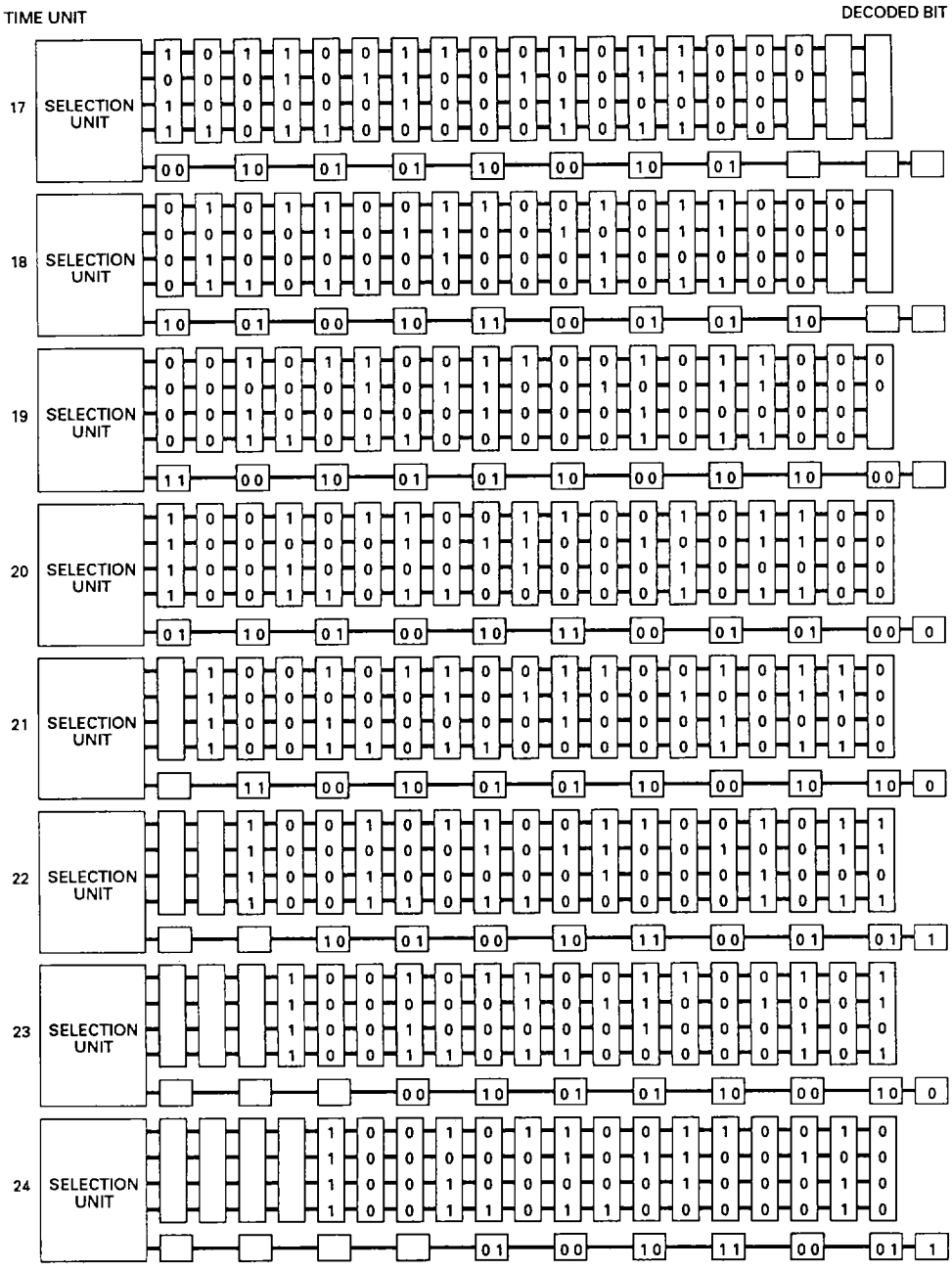
TIME UNIT
DECODED BIT
17 SELECTION UNIT
18 SELECTION UNIT
19 SELECTION UNIT
20 SELECTION UNIT
21 SELECTION UNIT
22 SELECTION UNIT
23 SELECTION UNIT
24 SELECTION UNIT

Fig. 7. (contd)

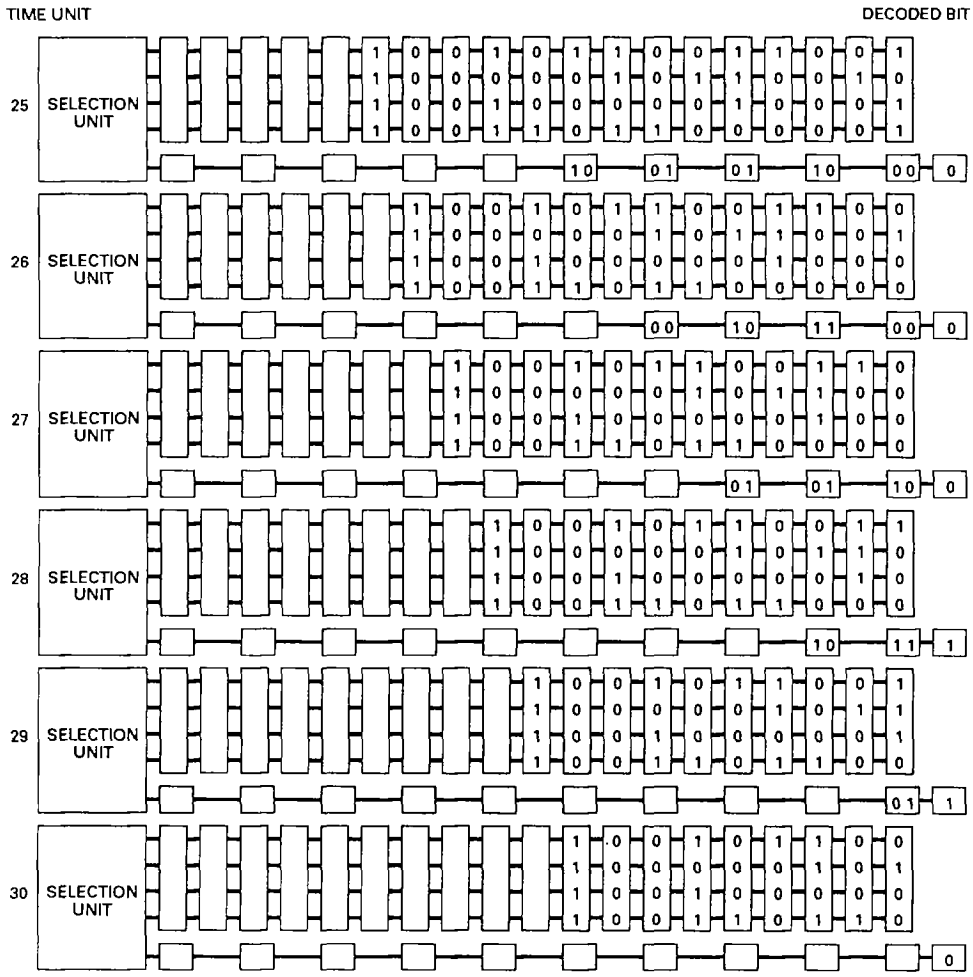
TIME UNIT
DECODED BIT
25
SELECTION UNIT
26
SELECTION UNIT
27
SELECTION UNIT
28
SELECTION UNIT
29
SELECTION UNIT
30
SELECTION UNIT

Fig. 7. (contd)